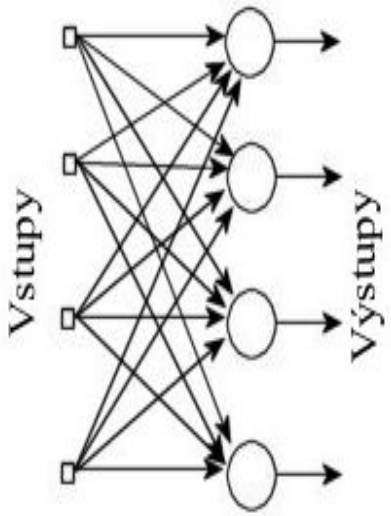
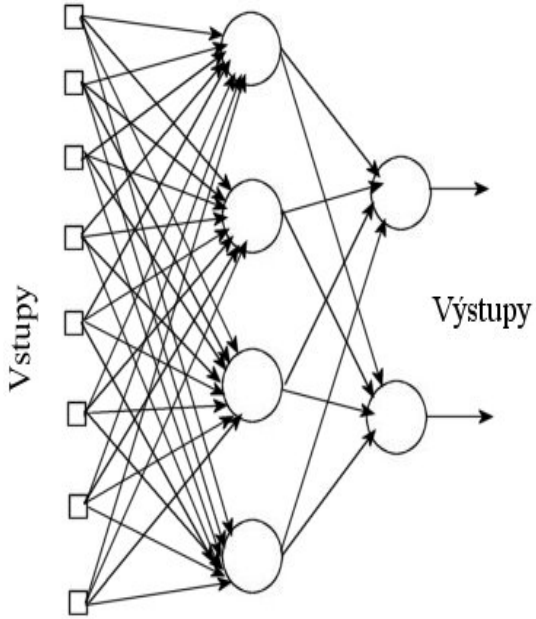
Layered ANNs

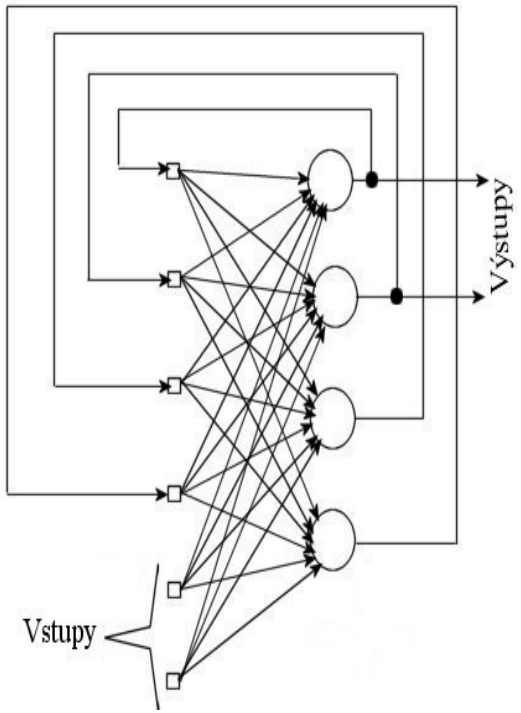


single layer



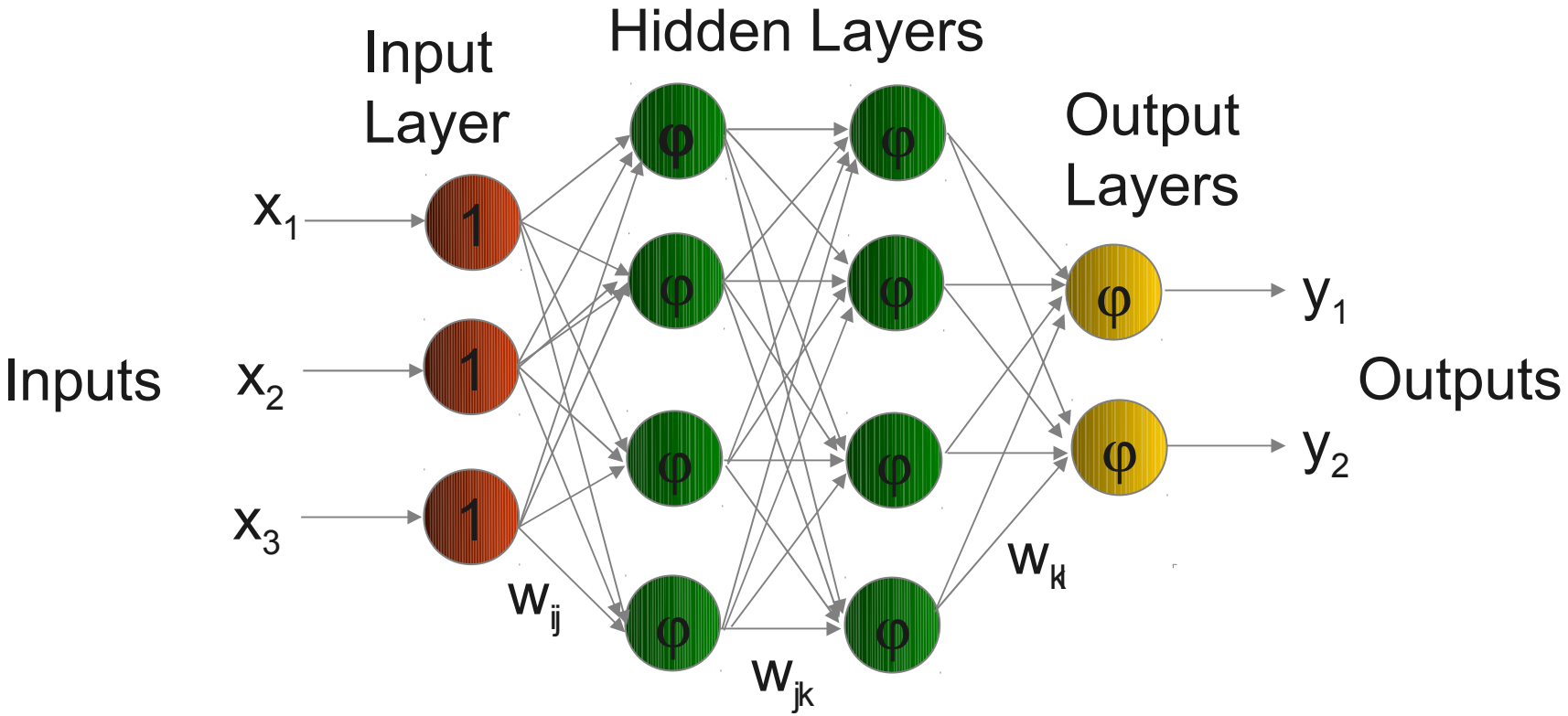
two layers

feed-forward networks

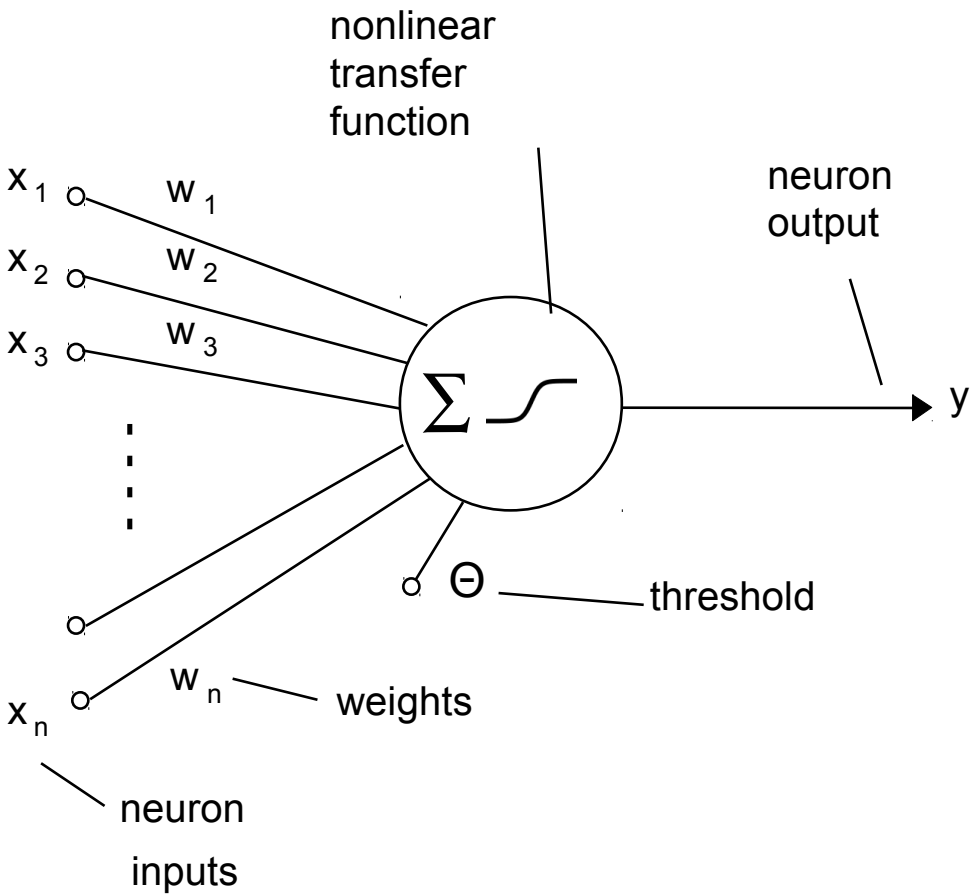


recurrent network

MultiLayer Perceptron (MLP)



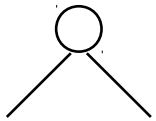
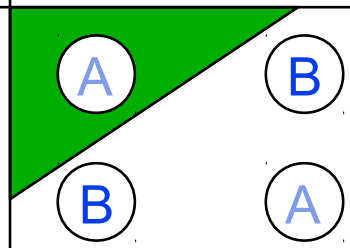
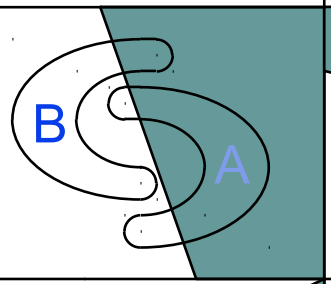
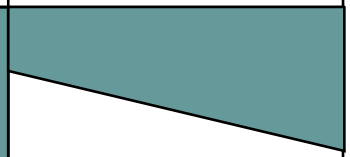
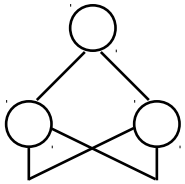
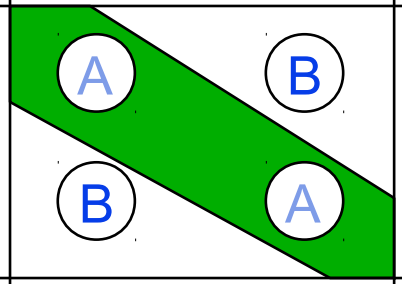
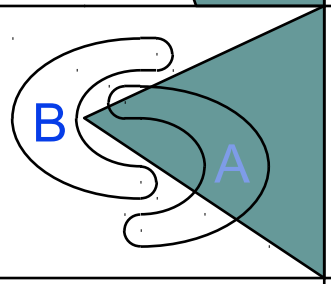
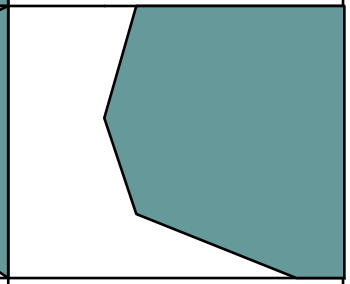
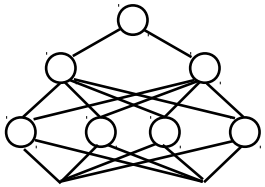
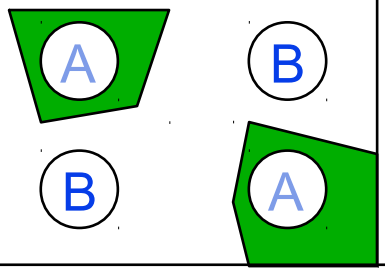
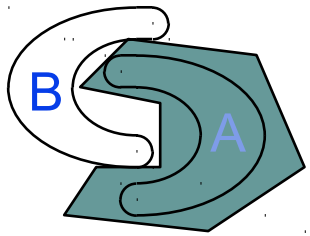
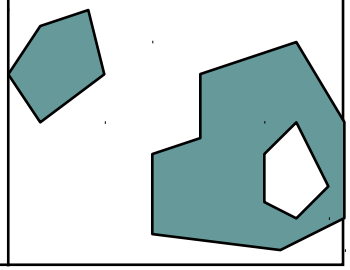
Neurons in MLPs



McCulloch-Pitts perceptron.

How Many Hidden Layers?
MLPs with Discrete Activation Funcitons

see ftp://ftp.sas.com/pub/neural/FAQ3.html#A_hl for overview

Structure	***Types of Decision Regions***	***Exclusive-OR Problem***	***Classes with Meshed regions***	***Most General Region Shapes***
Single-Layer 	*Half Plane Bounded By Hyperplane*			
Two-Layer 	*Convex Open Or Closed Regions*			
Three-Layer 	**Arbitrary** (Complexity Limited by No. of Nodes)			

How Many Neurons?

No one knows :(we have only rough estimates (upper bounds):

ANN with a **single hidden layer:**

$$N_{\text{hid}} = \sqrt{N_{\text{in}} \cdot N_{\text{out}}} \ ,$$

ANN with two **hidden layers**:

$$N_{\text{hid}-1} = N_{\text{out}} \cdot \left(\sqrt[3]{\frac{N_{\text{in}}}{N_{\text{out}}}} \right)^2 , \quad N_{\text{hid}-2} = N_{\text{out}} \cdot \left(\sqrt[3]{\frac{N_{\text{in}}}{N_{\text{out}}}} \right) .$$

You have to experiment.

Generalization vs. Overfitting

- When training ANNs we typically want them to perform accurately on new previously unseen data.
- This ability is known as **generalization**.
- When ANN rather memorizes the training data while giving bad results on new data, we talk about **overfitting (overtraining)**.

Training/Testing Sets

Random samples.

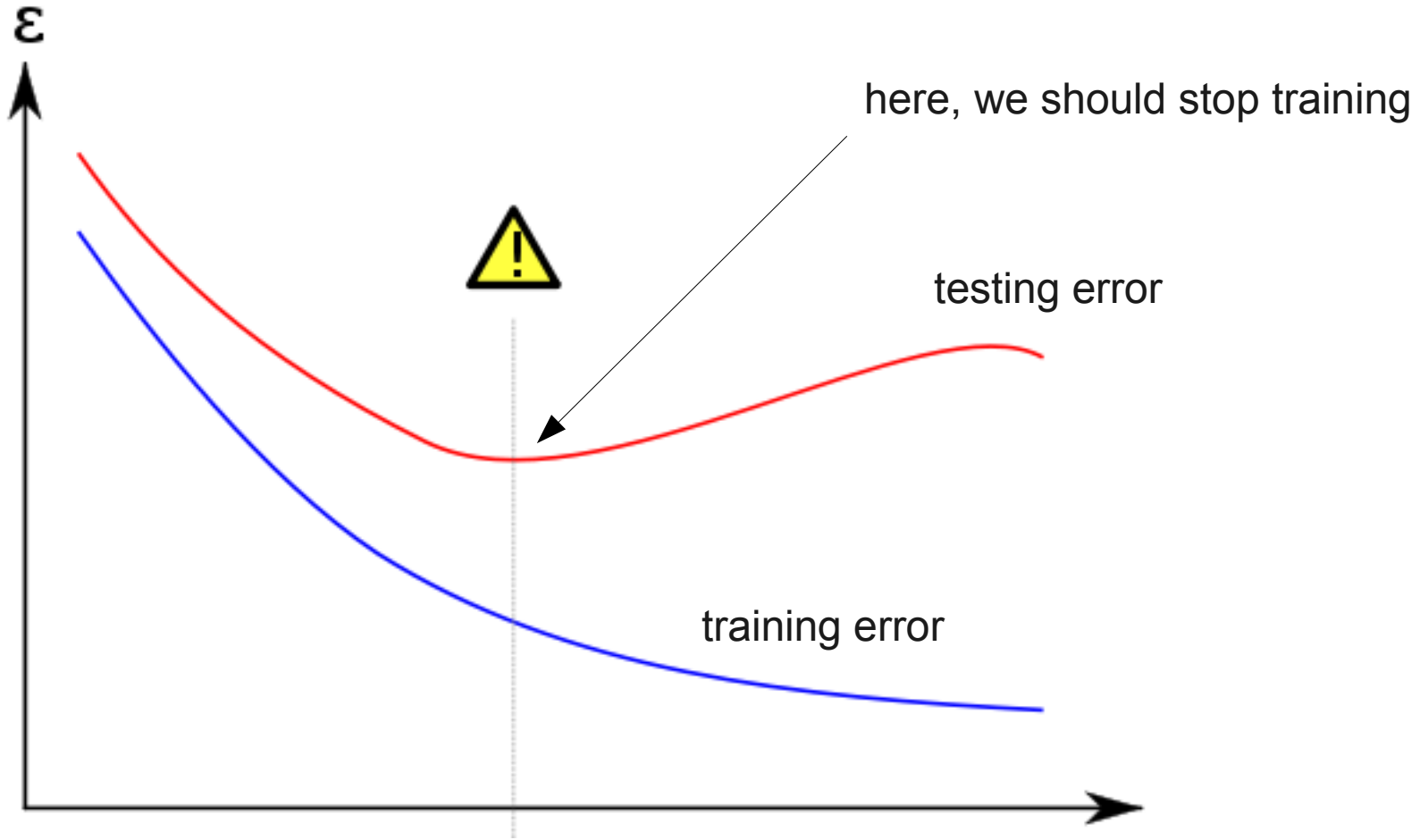
~70%

~30%

Used to train ANN model.

Testing set error
→ generalization

Overfitting Example

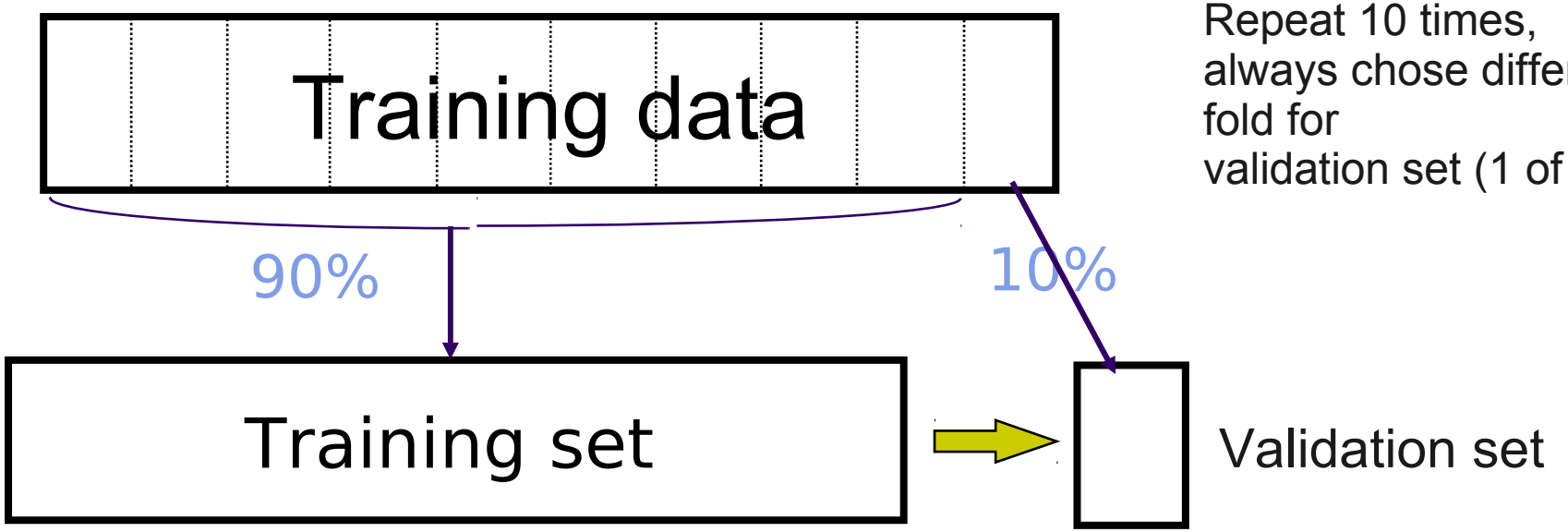


http://upload.wikimedia.org/wikipedia/commons/1/1f/Overfitting_svg.svg

k-fold Cross-validation

Example 10-fold cross-validation:

Split training data to 10 folds of equal size.



Repeat 10 times, always chose different fold for validation set (1 of 10).

Create 10 ANN models.
Choose the best for testing data.

The cross-validation error is the average over all (10) validation sets.

Suitable for small datasets, reduces the problems caused by random selection of training/testing sets

Backpropagation (BP)

- Paul Werbos,
- 1974, Harvard, PhD thesis.
- Still popular method,
- many modifications.
- **BP is a learning method for MLP:**
  - **continuous, differentiable activation functions!**

BP Overview

random weight initialization
repeat
  repeat
    choose an instance from the training set
    apply it to the network
    evaluate network outputs
    compare outputs to desired values
    modify the weights
  until all instances selected from the training set
until global error < criterion

ANN Energy

Backpropagation is based on a minimalization of ANN *energy* (= error). Energy is a measure describing how the network is trained on given data. For BP we define the energy function:

$$E_{TOTAL} = \sum_p E_p$$

The total sum computed over all patterns of the training set.

where

$$E_p = \frac{1}{2} \sum_{i=1}^{N_o} \left(d_i^o - y_i^o \right)^2$$

we will omit “p” in following slides

Note, ½ – only for convenience – we will see later...

ANN Energy II

The energy is a function of:

$$E = f\left(\vec{X}, \vec{W}\right)$$

$\vec{W}$ weights (thresholds) → **variable**,

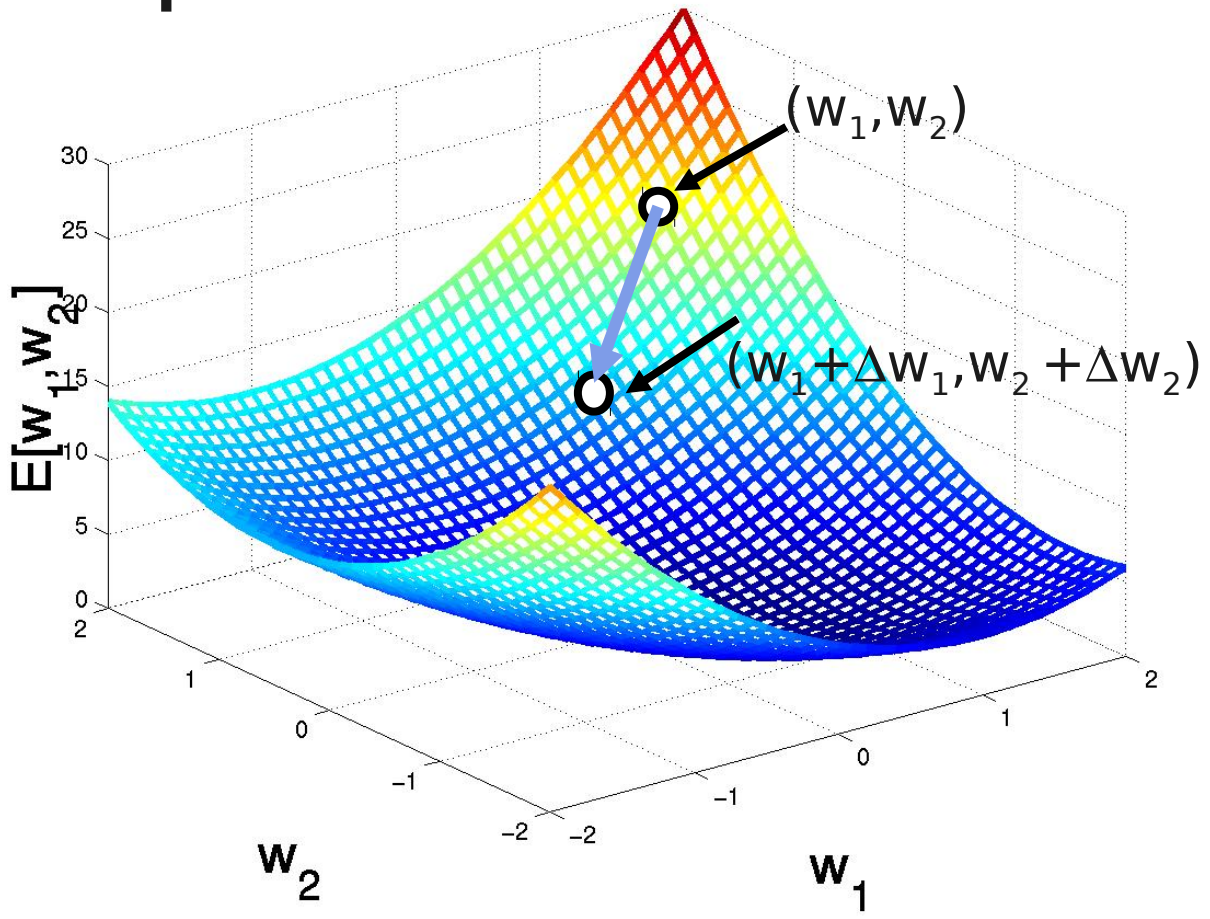
$\vec{X}$ inputs → **fixed (for given pattern)**.

Backpropagation Keynote

- For given values at network inputs we obtain an energy value.
- Our task is to minimize this value.
- The minimization is done via modification of weights and thresholds.
- We have to identify how the energy changes when a certain weight is changed by Δw.
- This corresponds to partial derivatives $\frac{\partial E}{\partial w}$.
- We employ a **gradient method**.

Gradient Descent in Energy Landscape

Energy/Error Landscape



Weight Update

- We want to update weights in opposite direction to the gradient:

$$\Delta w_{jk} = -\eta \frac{\partial E}{\partial w_{jk}}$$

weight "delta"

learning rate

Note: gradient of energy function is a vector which contains partial derivatives for all weights (thresholds)